Building High-Performance Linux Clusters, Sponsored by Appro

Writen by Logan G. Harbaugh, Networking Consultant for Mediatronics

WHITE PAPER / JUNE 2004

Abstract: This paper provides an overview as well as detailed information on the engineering approaches and the challenges of building a high-performance computing cluster. In addition, it presents comprehensive information focusing on Linux running on the AMD Opteron™ processor family and a real example of building and provisioning an Appro HyperBlade 80-node cluster with a total of 160 AMD Opteron processors at the AMD Developer Center.

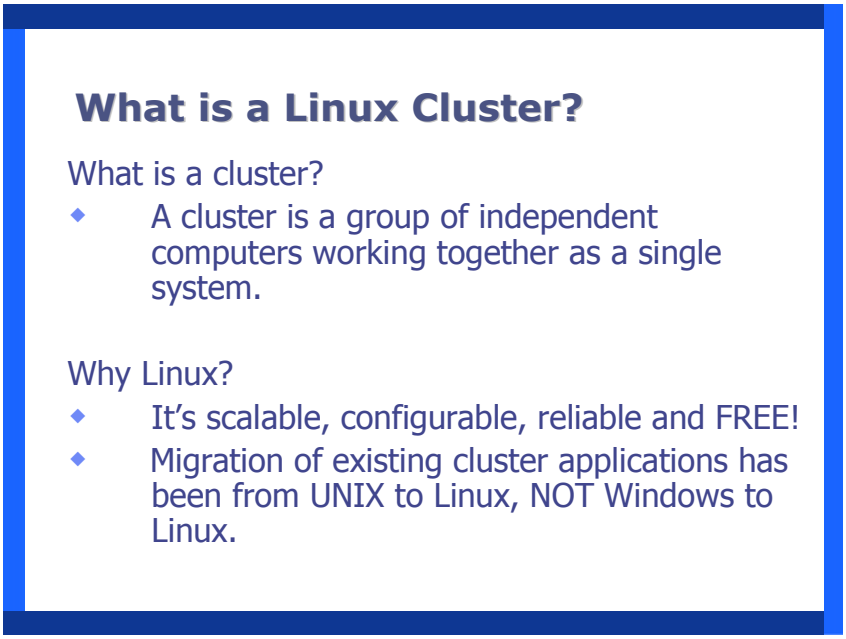
APPRO
HPC Cluster Solutions

Table of Contents

Introduction

Clusters built from off-the-shelf components are being used to replace traditional supercomputers performing trillions of calculations per second. Running Linux, these clusters are replacing supercomputers that cost significantly more, and allowing scientific institutions and enterprises to perform computations, modeling, rendering, simulations, visualizations and other sorts of tasks that a few years ago were limited to very large computer centers.

This white paper is intended to offer an explanation of computational clustering, specifically, clusters based on processors from Advanced Micro Devices (AMD) and Intel. While a single paper cannot cover every variation of clustering hardware and software possible, we will provide you with an overview of an HPC Linux Cluster and will thoroughly cover a specific Linux-based cluster used at the AMD Developer Center. This paper will provide enough details to get you started on cluster technologies.



A Brief History of Clustering

Clustering is almost as old as mainframe computing. From the earliest days, developers wanted to create applications that needed more computing power than a single system could provide. Then came applications that could take advantage of computing in parallel, to run on multiple processors at once. Clusters can also enhance the reliability of a system, so that failure of any one part would not cause the whole system to become unavailable.

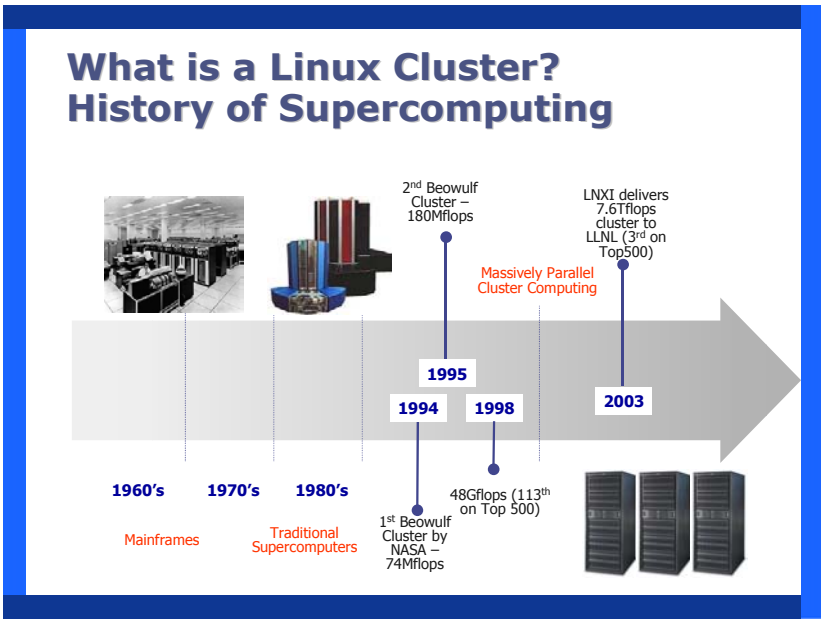
After the mainframes, mini-computers and technical workstations were also connected in clusters, by vendors such as Hewlett-Packard, Tandem, Silicon Graphics (SGI) and Sun Microsystems. These systems used proprietary hardware and proprietary interconnect hardware and communications protocols. Systems such as the Tandem Non-Stop Himalaya (acquired by Compaq, and now owned by Hewlett-Packard), provide automatic scaling of applications to additional nodes as necessary, as well as seamless migration of applications between nodes and automatic fail-over from one node to another.

The challenge with proprietary clusters is that the hardware and software tend to be very expensive, and if the vendor ceases support of the product, users may be marooned. Microsoft and Novell both proposed open clusters built on their respective Windows NT and NetWare operating systems and commodity

hardware. Although neither of these proposed clustering environments were deployed, the idea of using off-the-shelf hardware to build clusters was underway. Now many of the largest clusters in existence are based on standard PC hardware, often running Linux.

One of the first known Linux-based clustering solutions is the Beowulf system, originally created in 1994 by Donald Becker and Thomas Sterling for NASA. The first Beowulf cluster was comprised of 16 486 PCs connected by Ethernet. Today there are many organizations and companies offering Beowulf-type clusters, from basic toolsets to complete operating systems and programming environments.

The relative simplicity of code portability from older UNIX clusters to Linux is probably the biggest factor in the rapid adoption of Linux clusters.



Off-the-Shelf Clusters

At the time this was written, the largest supercomputer (measured in Teraflops capacity) was the Earth Simulator, as reported on the Top500 (www.top500.org) list. It is a mainframe computer built by NEC on proprietary hardware. A cluster of HP Alpha Servers at Lawrence Livermore National Laboratory ranked number two, and a cluster of Macintosh G5s connected with an Infiniband network was number three.

Given the advantages associated with low-cost, off-the-shelf hardware, easy, modular construction of clusters, open, standards-based hardware, open-source operating systems, interconnect technologies, and software development environments, it is to be expected that use of commodity clusters, especially clusters based on Linux, will continue to grow.

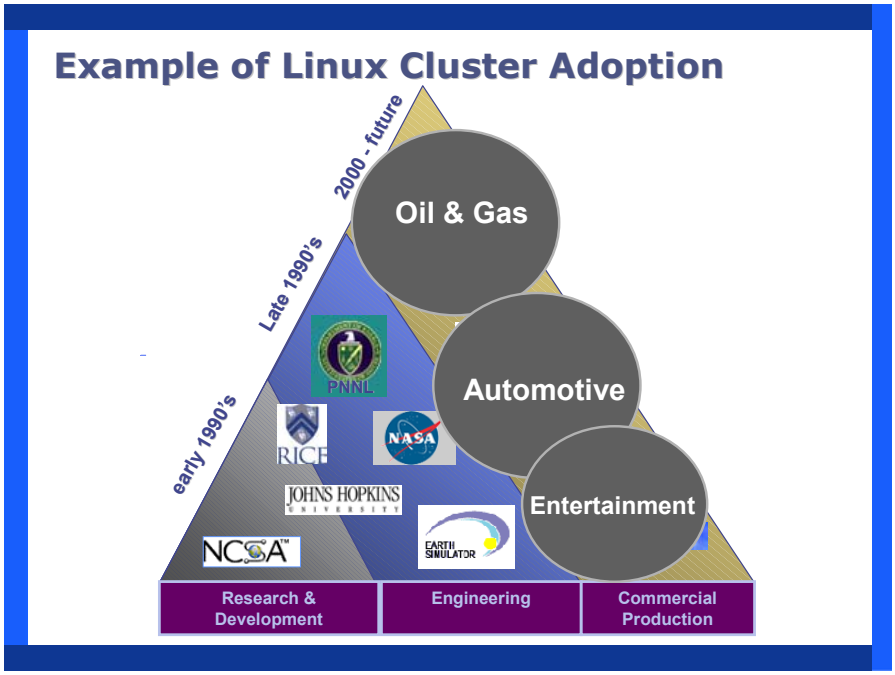
Clusters built with off-the-shelf hardware are generally AMD or Intel-based servers, networked with gigabit Ethernet, and using Infiniband, MyriNet, SCI, or some other high-bandwidth, low-latency networks for the interconnect; the inter-node data transfer network. Linux is becoming the cluster OS of choice, due to its similarity to UNIX, the wide variety of open-source software already available, as well as the strong software development tools available. Newsgroups such as linux.debian.beowulf, and comp.parallel.mpi are great places to ask questions, and there are thousands of web sites dedicated to clustering. A representative sample are listed at the end of this document.

High-Performance Linux Cluster Architectures

Clusters are not quite commodities in themselves, although they may be based on commodity hardware. Companies such as Appro can provide all the hardware necessary for clusters from four nodes to thousands, with integrated hardware management, integrated interconnect and networking support, pre-installed operating systems, and everything else necessary to begin computation.

A number of choices need to be made before assembling a cluster. What hardware will the nodes run on? Which processors will you use? Which operating system? Which interconnect? Which programming environment? Each decision will affect the others, and some will probably be dictated by the intended use of the cluster.

For example, if an application is written in Fortran 90, that will dictate the compiler to use. Since the popular GNU- compiler used for many clustered applications doesn't support Fortran 90, a commercial compiler will have to be used instead, such as one from the Portland Group (PGI). In fact, it may require separate sets of operating systems, utilities, libraries, compilers, interconnects, and so forth, for different applications.



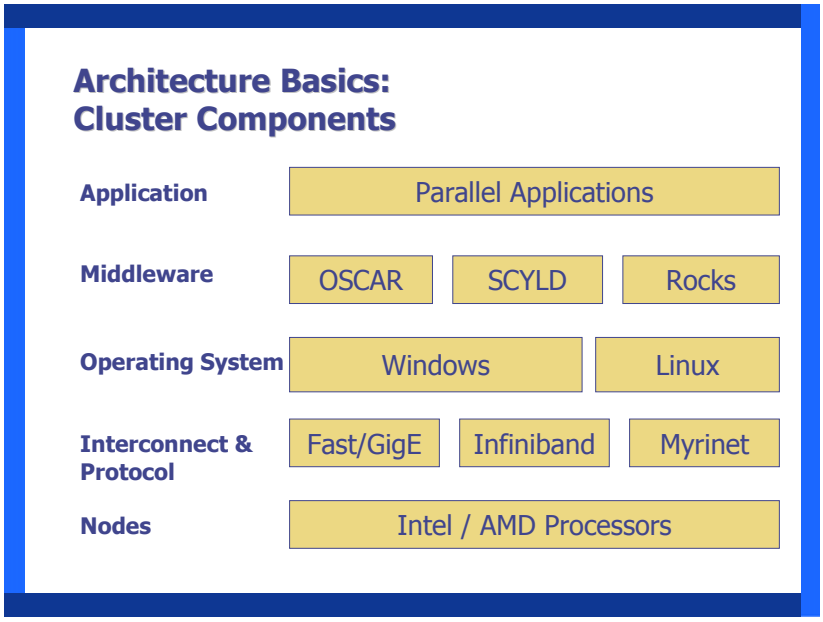
Picking the right cluster parts involves research.. A vendor may be able to help with the choices. Since application performance varies with hardware, count on substantial development time to research and evaluate the pieces of the cluster before deployment of the cluster application itself.

There are some basic questions to ask first. Will the application be primarily processing a single dataset? Will it be passing data around, or will it generate real-time information? Is the application 32- or 64-bit? This will bear on the type of CPU, memory architecture, storage, cluster interconnect, and so forth. Cluster applications are often CPU-bound, so that interconnect and storage bandwidth are not limiting factors, but this is by no means always true.

High-performance computing (HPC) pushes the limits of computing performance, enabling people in science, research, and business to solve computationally intensive problems, such as those in chemistry or biology, quantum physics, petroleum exploration, crash test simulation, CG rendering, and financial risk analysis. Over the years, HPC solutions have taken the form of large, monolithic, proprietary, shared-memory processing or symmetrical multi-processor (SMP) supercomputers (also referred to as vector processors) and massively parallel processing (MPP) systems from Cray Research, SGI, Sun, IBM and others, which utilized hundreds or thousands of processors.

HPC clusters built with off-the-shelf technologies offer scalable and powerful computing to meet high-end computational demands. HPC clusters typically consist of networked, high-performance servers, each running their own operating system. Clusters are usually built around dual-processor or multi-processor platforms that are based on off-the-shelf components, such as AMD and Intel processors. The low cost of these components have brought Linux clusters to the forefront of the HPC community.

HPC clusters are developed with throughput in mind as well as performance. Most HPC applications are too large to be solved on a single system. This is often the case in scientific, design analysis, and research computing. One solution is to build a cluster that utilizes paralleled software that breaks down the large problem into smaller blocks that run in parallel on multiple systems. The blocks are dispatched across a network of interconnected servers that concurrently process the blocks and then communicate with each other using message-passing libraries to coordinate and synchronize their results. Parallel processing produces the biggest advantage in terms of speed and scalability.



Node Hardware

The first piece of the puzzle is the node hardware. This will likely be either an AMD or an Intel processor-based server, but there are many choices to be made here, and they will depend on both your intended application and other factors, such as the physical area you have available for the cluster.

There are three form factors to be aware of – first, stand-alone PCs, second, rack-mount servers, and third, blade systems. Stand-alone PCs may be the least expensive, at least for the individual nodes, but they will use much more space, and the management of cabling and physical management (powering of/off, etc) will be much more difficult than with rack-mount or blade servers. Rack-mount servers are relatively dense, with 42 1U (a U is one rack unit, or 1.75" high) servers fitting in a standard rack cabinet. Simply buying 42 1U servers is only a start, though will also be needed – power strips, interconnect switches, the rack cabinet to put the servers in, and so forth. Blade systems are typically a little more expensive in initial cost, but they offer complete integration – the rack, Ethernet switches, interconnect switches, power delivery and management, as well as remote management of the individual nodes, and higher densities, with 80 or more dual-processor nodes fitting in a single rack cabinet.

The AMD Opteron™ processor with Direct Connect Architecture offers some advantages over the Intel Xeon processor. First, AMD64's integrated memory controller provides higher memory bandwidth. This feature results in memory access time that can be twice as fast for the AMD Opteron processor, so long as the operating system kernel supports the feature. The Intel Xeon processor, on the other hand, is a

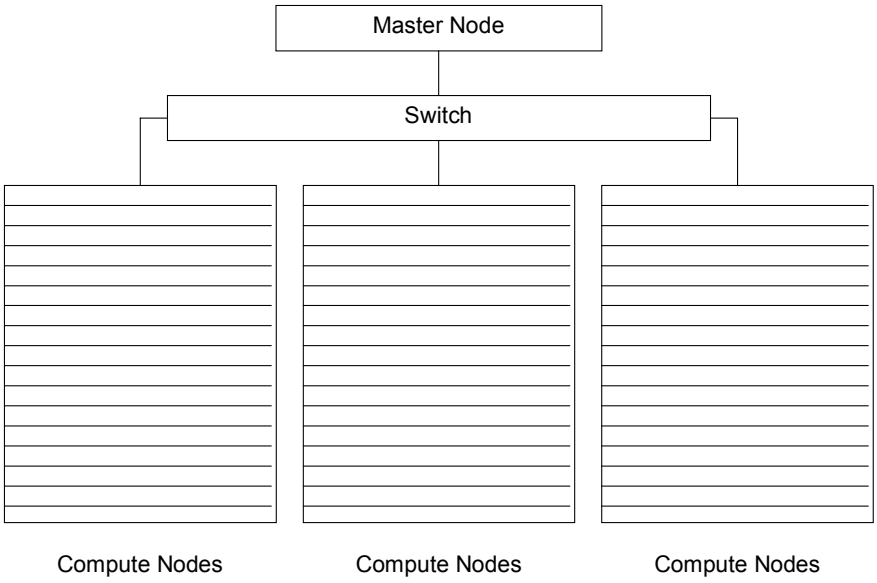
shared-memory architecture, in which the memory controller is on a separate silicon. In a multi-processor configuration, the memory controller is shared between the processors.

Another advantage is that AMD processors generally have a price/performance advantage, resulting in a lower total cost of ownership. This becomes more important when you are building a large cluster.

Other hardware factors will include the number and type of PCI, PCI-X® or PCI-Express™ expansion slots available. If the chosen node hardware does not have gigabit Ethernet or the correct interconnect, make sure there's a slot for the network adapter and one for the interconnect host bus adapter (HBA). It is a good idea to choose an HBA based on the type of application being developed and consider what type of PCI slot it uses (32-bit, 64-bit, 66, or 133 MHz, PCI-X or PCI-Express).

Some other things to consider, depending on the application, include: the type of memory architecture and memory speed, storage requirements (either internal, for the boot disk, or external for shared data), and availability of other ports.

There are usually two types of nodes, master nodes, and compute nodes. Master nodes, also called front-end nodes, are generally the only ones visible to the public network, and will need to run additional security and access control software, DHCP and DNS servers and other support software. They also typically have at least two network interfaces, one for the public network for external communication, and one for the private network to the cluster. Users can enhance the master nodes with larger hard disks, more memory, faster CPUs or extra network interfaces.



Power and Cooling Issues

Since space is usually at a premium, it is desirable to consider the most compact systems available. This generally becomes a balancing act between power and size. For instance, it is possible to squeeze about a dozen mini-tower PCs into a space two feet by three feet, if you stack them up. A standard seven-foot rack will hold 42 1U servers in the same space. A blade system, such as the Appro HyperBlade, can put 80 dual AMD Opteron processor-based servers/systems or dual-Xeon blades in the same space. Some blade servers, by using more compact (and typically less powerful) processors such as the Transmeta Crusoe, can put hundreds of CPUs in a single rack, though generally without the capacity for high-speed interconnects.

One issue that might not immediately come to mind when planning a cluster is power and cooling. Large clusters can be every bit as demanding as the old mainframes when it comes to power and cooling. At 60-100 watts in power consumption per processor, a large cluster can use substantial amounts of power. All of that power creates heat – an 80-blade, 160-CPU cluster can create more than three kilowatts per square foot of heat, which is more heat than is generated by an electric oven. HVAC systems in datacenters need to be carefully considered to support the heat density.

In addition to the servers, some other issues to consider are cabling for management networks, interconnects and power. Two Ethernet connections per server for 80 or more nodes require a large Ethernet switch, not to mention a substantial number of CAT5 cables. The cable count also depends on the network topology chosen for the cluster. A network can be built using a single switch, which offers single-stage latency. The switch must have sufficient number of ports, at least one for each node. When building a network for large clusters, choices of switches can be limited, and it may cost more for them. Alternatively, the same network can be built with several smaller switches in a cascading fashion. While smaller switches are more competitively priced, this multi-stage topology can lower network performance by increasing latency.

Node OS

While clusters can be built with different operating systems, Linux is a popular choice due to the low cost, a wide variety of available tools, and large body of existing research that has already been done on getting clustering to work. We will not address the unique requirements of clusters running on other operating systems in this paper.

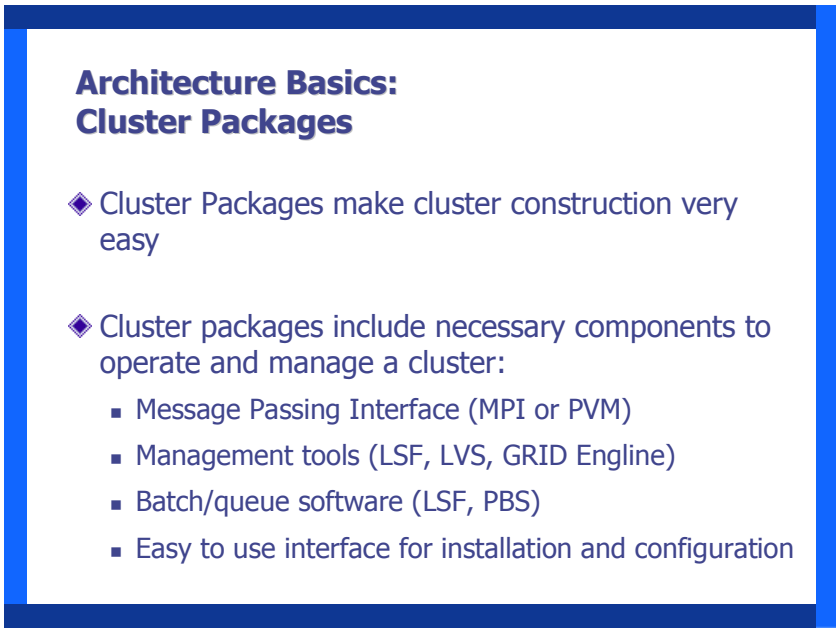
Even within Linux, there are several prominent choices, particularly Red Hat and SUSE (part of United Linux, which also includes TurboLinux and Conectiva), as well as a wide variety of smaller players including Mandrake, Debian, and many more. Choosing Linux may depend on the cluster software or what an IT staff may already be familiar with, or on features that are needed – for instance, the NPACI Rocks clustering distribution is integrated with the Red Hat Enterprise Linux 3 Advanced Workstation distribution. The stock Red Hat Enterprise Linux 3 SMP kernel does not include any NUMA (non-uniform memory access) features. The RHEL3 Update 1 SMP kernel supports NUMA memory affiliation. This is not a separate NUMA kernel, and therefore is not expected to perform as well as SUSE's NUMA kernel.

Appro offers Red Hat and SUSE and custom configurations using any desired operating system at additional cost.

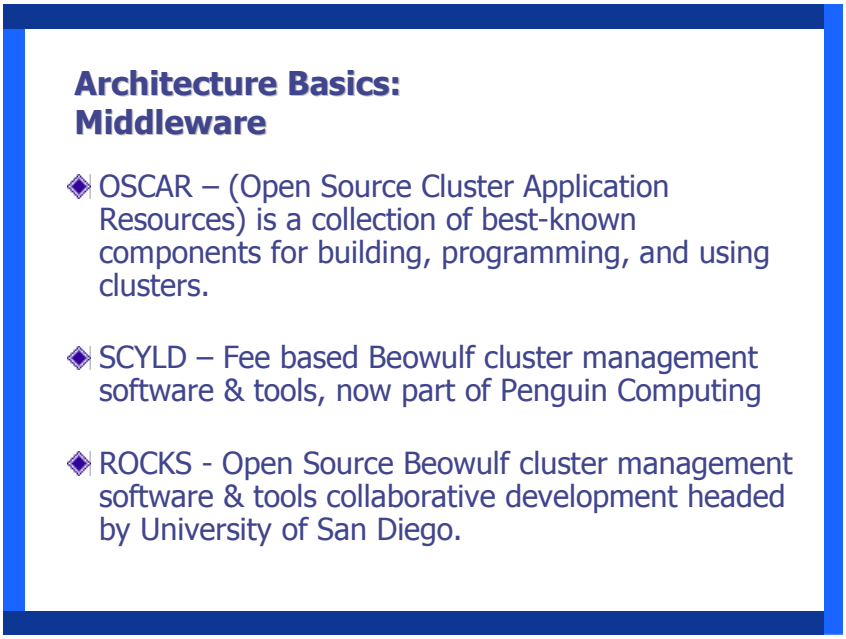
In addition to the OS itself, a large variety of ancillary software, such as compilers, libraries, configuration files, DNS and DHCP servers, SSH or RSH software and more will be necessary.It is important to develop and maintain software images for each type of node in the cluster. Each slave node has to be correctly configured with SSH keys, DHCP settings, etc. Debugging this is one of the hardest parts of getting a clustered application to work properly. Using pre-packaged cluster software will greatly simplify the configuration of various pieces of software needed for building a cluster.

Clustering Software

There are a variety of clustering software distributions available for Linux, ranging from compilers to complete clustering environments that include the operating system and all necessary drivers. Rocks, an open-source distribution created by the University of California at San Diego, and the commercial Scyld Beowulf operating system are two of the best-known complete distributions. As with the other choices, your choice of clustering software will depend on your other requirements.



MPICH is a freely available, portable implementation of MPI (Message Passing Interface), the standard for message-passing libraries. MPICH is maintained by Argonne National Laboratory (ANL) and Mississippi State University (MSU). Different versions of the MPICH may be necessary to support different compilers, and different versions of compilers may be necessary for compatibility with other parts of the clustering environment.



If there is only one application running on the cluster, it may be possible to create a single distribution and leave it in place. However, if, as with most mainframes, it is necessary to provide a service to a variety of departments, a flexible approach to provisioning and configuring the cluster is needed. To create a flexible clustering environment, investigate provisioning software, either open source software like SystemImager, or commercial software such as Norton's Ghost or Open Country's OC-Manager, or deployments using scripts or other open source tools.

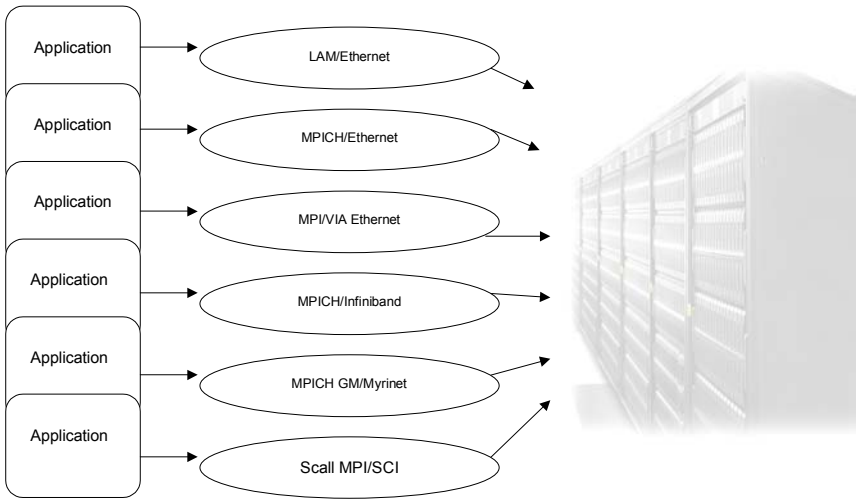
Commercial products, such as the Scyld Beowulf clustering environment, provide a complete system for creating clusters, including an MPI library, scheduler, and monitoring services. It can distinguish between installing a master node and compute nodes, and it includes programming tools and more. While there is an upfront cost to these solutions, it may be offset by shortened installation, configuration and development times. Plus commercial-grade software support comes with it, which is something that open source software may lack.

Today, the Message Passing Interface (MPI) dominates the market and is the standard for message passing. Although MPI is common for most parallel applications, developers are faced with a challenge; virtually every brand of interconnect requires a particular implementation of the MPI standard. Furthermore, most applications are statically linked to the MPI library. This raises three issues. First, if you want to run two or more applications on your cluster and some of them are linked with different versions of the MPI implementation, then a conflict might occur. This inconsistency is solved by having one of the application vendors re-link, test, and qualify their application for the other MPI version, which may take a significant amount of time.

Second, evolving demands from applications or errors detected and corrected in the MPI implementation can force one of the applications to use a newer version. In this case, the previously mentioned inconsistency may result. The third issue to watch for is upgrading the interconnect to a different kind, or evaluating the possibilities of doing so. Let's say Gigabit Ethernet has been chosen as the interconnect, but it is observed that the TCP/IP stack imposes overhead that restricts the scalability of the application. To switch to an MPI able to take advantage of more efficient and lean protocols, such as Remote Direct Memory Access (RDMA), the application vendors may be approached for help with your upgrade or evaluation. This may be an obstacle to major improvements that could be gained from newer more innovative communications software, or the general interconnect hardware evolution.

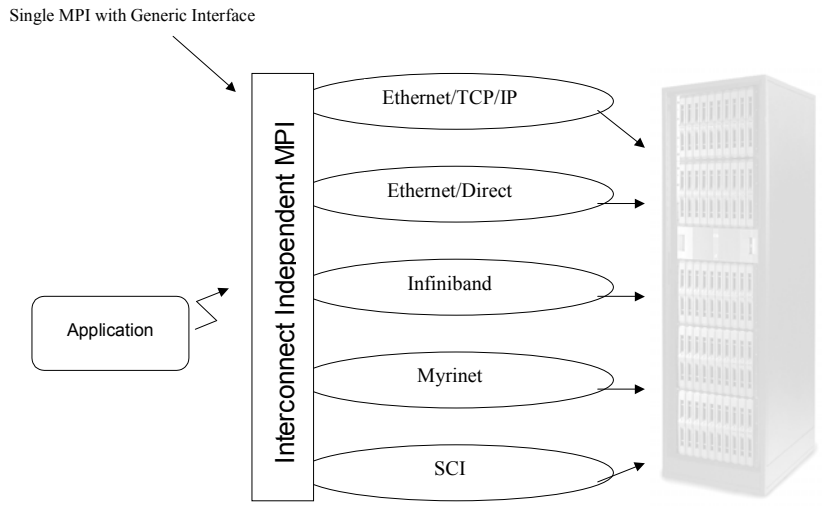
Another approach – one that avoids the above issue – is dynamic binding between the application and MPI middleware and between the MPI middleware and device drives for various types of interconnects. This way, the MPI implementation and application can evolve independently, because the application can exploit the benefits from different interconnects or protocols without changing or re-linking the applications. (Refer to the two pictures in the next page.)

Traditional MPI solution architecture



Traditionally, parallel MPI applications are statically linked to a particular MPI implementation, supporting a single interconnect type. Running the application on a different interconnect is therefore cumbersome as it requires re-linking the applications to a different MPI implementation which supports the specific interconnect.

Single MPI with interconnect interoperability



Advanced MPI implementations are linked dynamically to the parallel MPI application, which supports a run-time selection of the interconnect. Switching from one interconnect to another does not require re-linking and therefore is simply a matter of running the application on another interconnect.

Assembling the infrastructure is the simplest part of creating a cluster - although ensuring that the integrated whole of a cluster works together is not trivial, the task is still much simpler than developing applications to run on the cluster.

Applications must be cluster-aware – able to communicate thru MPI for the distribution of processes to multiple nodes and to collect the results. This is not as simple as re-compiling an existing multi-threaded application. In addition, the number of nodes or processes is typically not dynamic, although process queuing and job management applications are available, which can be used to allocate processes to nodes as they become available.

In a typical scenario, an application is compiled using the GCC compiler and MPICH, which "wraps" commands to add the cluster-aware functionality. MPICH uses a *machines* file to keep track of the nodes in the clusters, and each version of MPICH has a slightly different nomenclature for the file. Some list the private network IP address for each node in the cluster, some with just the IP address of each node, some with the form *IP Address*:1, *IP Address*:2 for each CPU in a node, or just a listing of each IP address one time for each CPU. The MPICH distribution will have an example file with the expected nomenclature.

The "mpirun" command starts a clustered run of code. MPI can identify all the nodes in a cluster, where the libraries are stored, and logs into each system once for each process/CPU (normally twice per blade). In order to streamline the login process, it may be desirable to create a limited user with no password on the compute nodes.

Scientific applications that use Fortran may require using the PGI compiler rather than the GNU compiler, since GNU doesn't support Fortran 90. PGI Workstation 5.1 supports both Fortran 90 and previous versions.

On AMD Opteron processor-based systems, the AMD Core Math Library (ACML) 2.0 provides a suite of math functions optimized for the AMD Opteron processor. This library is open source and downloadable from AMD (www.developwithamd.com/acml).

Time synchronization is critical for clustered applications, so ensure that NTP is included in the compute node image.

Provisioning for node failure is important. Simple cluster applications allocate nodes statically – the *machines* file is a hard-coded list of available nodes, and by default, all nodes are used. This means that if any one node fails, the entire clustered application may also fail. An application or script can be written to ping each node and remove nodes from the *machines* list as necessary, but the default is also that applications expect a set number of nodes. If there is a node failure, the application run can fail if the number of nodes decreases, so a mechanism to bring up a hot spare to maintain the total number of nodes is also desirable. A queuing system can mediate the number of CPUs available.

While applications are running, a status monitor is a very useful thing to have. Ganglia is an open source cluster monitoring application that puts a small footprint daemon on each node, and shows CPU, memory and network utilization, and other statistics for each node.

Interconnects

There may be as many as four networks associated with a cluster. All clusters will have a private network, used for basic communication between nodes. Most will also have a public network, which will connect with the rest of the company, and possibly the Internet and the rest of the world. It is important to make sure there is a firewall (NAT, if nothing else), between the public and private networks. In addition, clusters may also have interconnect networks and storage area networks (SANs).

The public and private networks will probably be Ethernet, whether 10/100 or Gigabit will depend on the requirements for distributing node images and starting and stopping jobs, etc. Interconnect networks are used to pass operational data between nodes of the cluster, and the requirements for interconnects vary widely with the type of application running on the cluster. Many cluster applications are CPU-bound, which is to say that processing power is the bottleneck, rather than network bandwidth. In this case, the interconnect might be 100 Mbit/sec or Gigabit Ethernet without affecting cluster performance. On the other hand, if the cluster application requires passing large amounts of data around, then a higher speed interconnect will make a significant difference.

Popular interconnects include Gigabit Ethernet, Infiniband, SCI (scalable coherent interface), and MyriNet. Gigabit Ethernet is the least expensive interconnect on a per node basis, but not necessarily the cheapest in cost per operation. In a test done at the AMD Developer Center, a 64-node cluster with AMD Opteron processor Model 244 nodes running Gigabit Ethernet produced 1,012 million operations per second and cost $100,000, resulting in a price of $1 per 10K operations per second, while a 16-node cluster with AMD Opteron processor Model 246 nodes running Infiniband 4x produced 2,424 million operations per second, and cost $50,000, resulting in a price of $1 per 48K operations per second. This application was network-bound. If an application is CPU-bound, a high-speed interconnect may provide less of an advantage.

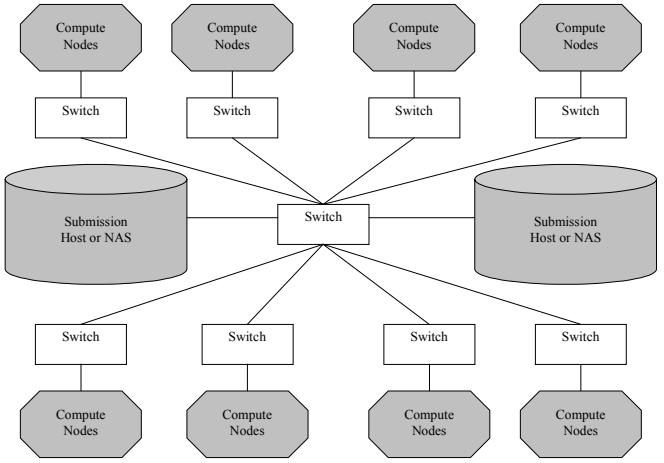
In addition to speed and cost per port, other considerations for interconnects may be the number of ports available on switches, whether drivers are available for the version of Linux and the kernel version you want to use, and availability of hardware from multiple sources.

Cluster Interconnect Performance

Interconnect	Latency (useconds)	Bandwidth
100BASE Fast Ethernet	80	12 Mbps
1000BASE Gigabit Ethernet	~50	125 Mbps
Myrinet*/32-bit	7	80 Mbps
Myrinet*/64-bit	7	160 Mbps
InfiniBand Architecture (x4)	5.5	10 Gbps

Storage area networks might be desirable separately from both the public/private networks and the interconnect network, depending on the application. Video processing clusters, used for movie special effects, often have both high-speed interconnects and fibre channel SANs. If this is the case for your application, you will want to look for a server or blade that can support multiple host bus adapters (HBAs).

Storage Area Networks



The picture above is a sample topology of how a cluster can be networked together in multiple stages, using one core switch and several edge switches. Storage is directly attached to the network.

Deployment

Deployment of images to all the nodes in a cluster can take a number of approaches. NCAPI Rocks treats the entire OS image as a single entity, and re-deploys the whole image as necessary, using PXE boot. Run a DHCP server on the master node rather than assigning static addresses, to allow for ease of moving a node from one cluster to another or replacing nodes.

If a cluster is using one OS image, variations due to different compilers, libraries and configuration files can be handled using NFS mounts, but using substantially different OSes, it will be simpler to re-deploy disk images, either using shell scripts, SystemImager, or Symantec's Ghost. Commercial products require an upfront cost, but are much easier than managing revisions to scripts and images, which over time could result in cost savings.

Cluster Management Overview

An 80-node, 160-processor cluster requires a lot of hardware. Imagine 80 PCs stacked up in a pile, with keyboard, mouse and monitor cables, network cables, and power cables running to each one, plus KVM switches, Ethernet switches and power strips. Even a couple of racks of 1U servers have the same issues. Power management systems such as the APC MasterSwitch, which allows remote, web-based control of power, and serial console servers for access to the serial terminal of each server, and PCI-based AMI server management cards for hardware monitoring, can be added and eventually get to the kind of functionality that comes with the HyperBlade. But it can be expected to take a lot of time and money to get there.

Integrated Cluster Solutions – The Appro HyperBlade Cluster Series

The Appro HyperBlade Cluster Solution provides an extremely flexible and manageable way to create high-density clusters. It is designed to scale, with little effort or re-configuration, from a few nodes to 80 dual-processor nodes in a single rack (twice the density you can get with 1U servers), using the Mini-Cluster, the Mid-Cluster, or the Full Cluster solutions.

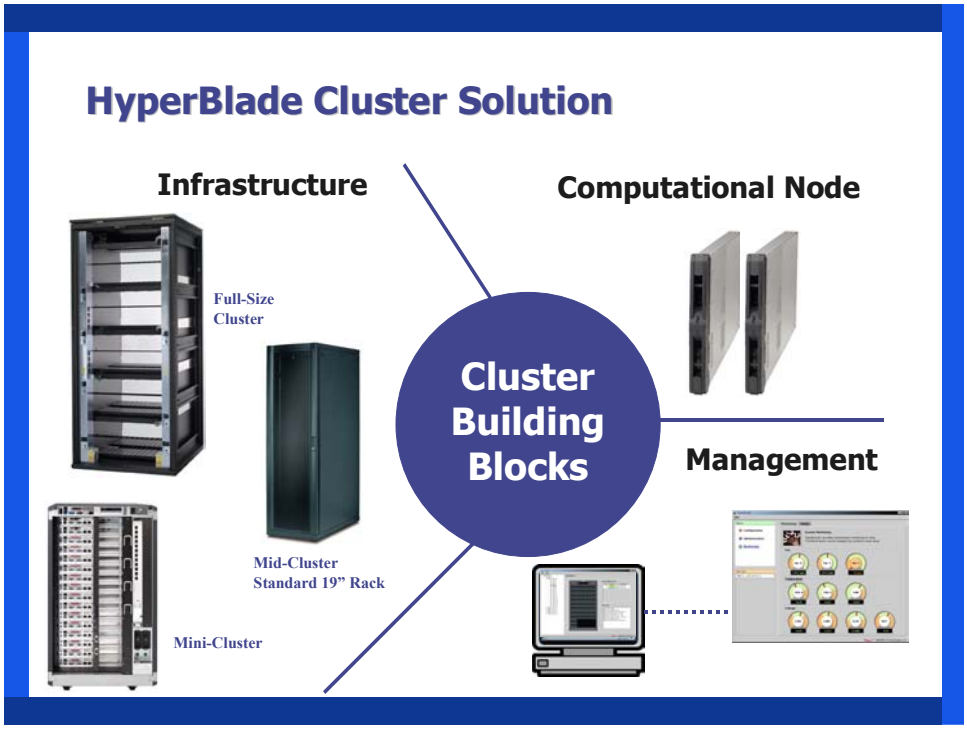
These cluster solutions provide some of the fastest, most powerful CPU cores available anywhere, ready to run your Bewoulf clusters, with applications in scientific visualization, financial modeling, digital rendering, seismic analysis, and many others that requires the highest floating-point and memory bandwidth performance.

The Mini-Cluster is a small standalone rack that supports up to 17 Appro HyperBlade server nodes, which can be single or dual processor: AMD Opteron processor, Xeon, Pentium® 4 or Itanium 2 blades. The Mid-Cluster supports up to 50 nodes in a standard 19" wide, full-height rack, using five sub-racks each holding 10 blades. The Full Cluster is a large rack that holds 80 HyperBlade server nodes in five sub-racks each housing 16 blades. By fully populating one HyperBlade rack cabinet with dual 2GHz processors, a peak computation capability of 640 GigaFLOPS, or a power density of 72 GFLOPS per square foot may be reached.

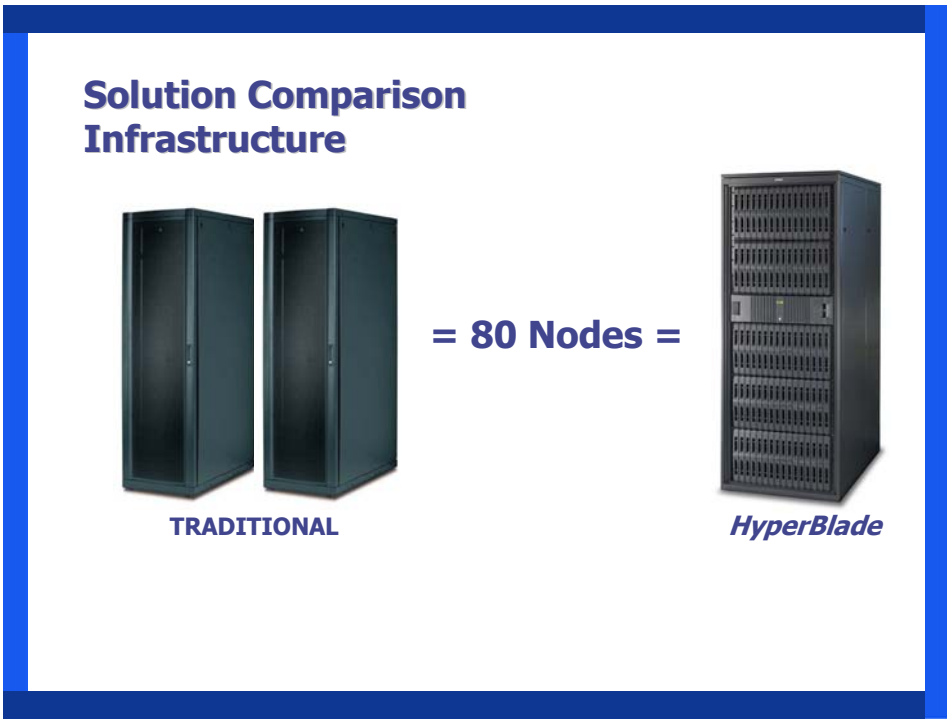
Since the blades are interchangeable between all three cluster systems, it is possible to start with a few nodes in a Mini-Cluster and expand to a full cluster of 80 nodes, without having to buy all new hardware at each step. Further, nodes with different processors can be mixed in the same system, allowing several types of clusters to be created for specific purposes, while retaining the manageability and compact form factor of the HyperBlade system.

Each node is an independent server, slightly smaller than the usual 1U server. It can, if desired, be used separately from the HyperBlade racks, since all usual connections – keyboard, mouse, monitor, dual Gigabit Ethernet ports, USB ports, serial port, etc., are available on each blade. The blades also include a full-length 64-bit 133 MHz PCI-X slot, so any desired interconnect can be added – Infiniband, Fibre Channel, MyriNet, Dolphin (SCI), etc.

Picture: HyperBlade server based on Dual AMD Opteron™ processors



Infrastructure, compute servers, and management tools are the three main building blocks for a cluster.



One HyperBlade cluster saves space by providing the same number of servers as two typical-sized cabinets, resulting in a 66 percent density increase.

Appro Cluster Management: Appro BladeDome and Blade Command Center

In addition to managing power, cooling, network cabling and so forth, the process of powering nodes on and off, booting, configuring BIOS, and managing each node of a cluster becomes an ever greater challenge with large clusters. The Appro Cluster Management solution, consisting of the Appro Blade Command Center and BladeDome software helps to simplify these management complexities.

BladeDome is a hardware monitoring and management solution provided by hardware and software components. Each node in a HyperBlade cluster is connected to the Blade Command Center, which provides both in-band and out-of-band management from the BladeDome web-based console and command-line interface. A single secured Java GUI shows blade status for all blades - power on or off, CPU temperatures, fan speeds, voltages and other hardware status. It can also provide control of each blade, both hard and soft reboot, power on and off, and a serial terminal connection that starts at boot up, allowing full access to BIOS configuration and HBA configuration, as well as remote access to the console of each server. With the serial console feature, you are provided with virtual KVM (text-mode) connectivity to any compute node, viable from POST, to watching kernel-loading messages, to terminal access of the OS for each node. In other words, BladeDome offers environmental monitoring and node control, all within a single utility.

The Blade Command Center communicates with each HyperBlade compute node via a unique cable-less connector, connected to custom electronics mounted on the back of each HyperBlade sub-rack. One subrack connects to the next in a cascading fashion, and all subracks are then connected to the Blade Command Center. The Blade Command Center serves data and accepts commands from a Java GUI client. It is also possible to control the cluster using the command line interface by telnet or ssh into the Blade Command Center.

Solution Comparison
Server and Cluster Management

cyclades

Digi Connectware™

Vs.

TRADITIONAL

Appro BladeDome and Blade Command Center

	Traditional – 80 Nodes Cluster	HyperBlade - 80 Nodes Cluster
Cabinet	2X APC NetShelter VX 42U	HyperBlade with BladeDome
Console Management	2X Console Server	Integrated
Power Management	5X APC Master Switch VM AP9229	Integrated
Server Monitoring	AMI Server Management Card	Integrated

Appro BladeDome - hardware monitoring and control of each blade and Appro Blade Command Center - secure web console.

AMD Developer Center and Appro HyperBlade Cluster

The AMD Developer Center was created to assist software developers and customers on porting from 32-bit processors to AMD64 technology, and benchmarking their software in a secure environment on AMD processor-based systems. Customers can use the systems remotely through a secure VPN connection or they can be used onsite at One AMD Place in Sunnyvale, California.

APPRO is pleased to work with the AMD Developer Center to provide HyperBlade Supercomputing Cluster benchmarking opportunities based on the AMD Opteron processor. This is an important value-added service to Appro's customers, who are able to benchmark and test the AMD Opteron processor-based systems and selected software prior to making a final purchasing decision. The creation of the AMD Developer Center demonstrates that AMD is committed to working with software developers and cluster partners to create tools that will help drive customer success.

A Real-World Example: AMD Developer Center

This real-world example will document the process that AMD's Developer Center Manager, Justin Boggs, and his staff used to create, install and provision an 80-node cluster using Appro's HyperBlade hardware. The example also provides some tips on best practices.

The "Niobe" cluster is installed at the AMD Developer Center. It is an 80-node Appro cluster using dual AMDOpteron processor Model 246 blades, with 2GB RAM per CPU, for a total of 160 processors and 320GB RAM. The cluster has 64 nodes connected by Infiniband – for tests that need a high-speed interconnect – the rest are held in reserve or used to create smaller secondary clusters.

Picture: The Niobe cluster has an interesting plenum built onto the back of the Appro cabinet.

The room the cluster is in was not originally built as a datacenter, and the air conditioning ducts were not designed to direct inlet air on the cluster. AMD built a plenum on the back of the HyperBlade cabinet that exhausts air out the back, and this also aids in drawing cool air from the front.

The blades are managed with Appro's BladeDome System Management tools. According to Boggs, BladeDome provides a very convenient single console for administering hardware, and the serial console redirection also makes it possible to access command-line consoles on each node if desired for debugging.

There are two styles of clusters in use at the AMD Developer Center – the San Diego Supercomputer Center's ROCKS and a home-built cluster based on SUSE Linux Enterprise Server 8 SP3 AMD64 (kernel SMP 2.4.21-143), the GCC 3.3 Compiler, The PGI 5.1 compiler, the MPI message passing interface and InfiniMPI 2.1 (based on MVAPICH).

ROCKS uses RedHat Enterprise Linux 3.0, with the Sun Grid Engine 5.3 as the default batch scheduling system, and PXE for deployment. For SUSE nodes, the lab uses Symantec Ghost Enterprise and BOOTP to distribute images.Ghost does not yet support the Reiser file system, which is the default file system with SUSE Linux Enterprise Server. This means that deploying images takes 2 hours with 64 nodes, vs. 10-15 minutes when using ext2 or ext3 file systems.

Several different MPICH compilers are used. Different versions of both the GCC and PGI compilers are used to ensure version compatibility and test stability issues across all available versions.

The MPIRUN command starts a clustered run of code. MPI can identify all the nodes in a cluster, where the libraries are stored, and logs into each system once for each process/CPU (normally twice per blade).

A username is created for the cluster's private network to allow for MPI processes to run. SSH provides the login mechanism for MPI as it scales better than RSH for large clusters.

A common set of shared files is normally shared via a mount, although some clustered applications that result in high network bandwidth may require that each blade have its own copy. The common files include the binaries and libraries for each test. Usually the network and I/O subsystems are the limiting factors in a shared network file system. If it is a problem, a script distribution of the shared files to each node is used.

The cluster setup uses a machines file that has a listing for each node in the cluster, some with just the IP address of each node, some with an IP:1, IP:2 for each CPU in a node, or just a listing of each IP address one time for each CPU. Each MPICH version has an example file with the expected nomenclature.

The PGI Workstation 5.1 compiler is specifically for Fortran 90, which is commonly used for math/scientific applications. GCC does not support Fortran 90. The AMD Core Math Library, both version1.5, and the new version 2.0 released 4/5/04 are also used for tests. The math library is open source, downloadable from AMD.

Infiniband is not yet natively supported, which means that IP addresses for Infiniband nodes aren't assigned until the end of the boot process. This also means that assigning the IP address for the Infiniband nodes cannot be done through DHCP, but must be scripted.

A couple of tips from the development team on writing code: If writing original code, make the application fault tolerant – ensure that a node failure won't take the whole clustered app down. The AMD lab reserves 10 percent of nodes for hot spares. Plan on lots of development/debug time. In our experience, SUSE Linux is a more mature, full-featured OS that offers solid 32- and 64-bit application support across hardware platforms

The following is a script that the team wrote to fork out commands to all the nodes in a cluster. It is actually generic, but useful for sending out the same command to each node. For example, this can be done to do an "uptime" on every node to see what the load is and to check to make sure every node is up by typing:

fork "uptime"

The 'machines' file needs to be a text file listing of all the systems in the cluster.

niobe-0:~ # more /usr/local/bin/fork

```
#!/bin/csh
foreach host ( `cat /etc/machines` )
echo $host - $1
ssh $host $1
end
```

The “front-end” or master node is set up with a NAT firewall, so that public network cannot directly access the compute nodes. This reduces the security risk associated with the compute node logins not having passwords.

Listing of Compiler and Tool Versions used at the AMD Developer Center:

GNU Compiler Collection (GCC): gcc; g++; g77; versions 3.2 and 3.3 optimized for 64-bit

PGI Workstation 5.1

Optimized Fortran 77/90, C/C++

Compuware SoftICE supports 32-bit and 64-bit AMD Opteron processor

AMD Core Math Library 1.5 and Core Math Library 2.0, jointly developed with the Numerical Algorithms Group (NAG), which includes Basic Linear Algebra Subroutines (BLAS) levels 1, 2 and 3, a wide variety of Fast Fourier Transforms (FFTs), and Linear Algebra Package (LAPACK). It is available for commercially available OSes.

ld - GNU linker.

Handles 32-bit and 64-bit code.

Invoking directly is discouraged; linking should be performed thru gcc/g++/g77.

Default: Produce elf64-x86-64 64-bit binaries.

"-m elf_i386“: Produce 32-bit i386 binaries

Many of the tools in the GNU C Compiler suite have new switches for AMD64 technology use, so that one tool can compile 32-bit or 64-bit code.

“as” - GNU assembler. Handles 32-bit and 64-bit code.

“--64”: Default, assemble x86-64 assembly into 64-bit code.

"--32”: assemble i386 assembly into 32-bit code.

“gcc/g++” - GNU C/C++ compiler. Handles 32-bit and 64-bit code.

“-m64”: Default, compile to 64-bit, x86-64 code.

"-m32“: compile to 32-bit, x86 code.

Picture: The rear of the HyperBlade Full Cluster show how neatly everything comes together.

The HyperBlade racks include integrated power deliver systems, cooling systems, and Ethernet switches, SMC Gigabit Ethernet by default, with Cisco or Foundry switches optionally available. The Mini-Cluster uses one 220v three-phase connection to consolidate power for all its blades, while the Mid-Cluster and Full Cluster use two 220v three-phase connections. There is also space in the cabinet for interconnect switches, which Appro will install as desired. Cabling is noticeably simplified, when compared with a rack full of 1U servers, and the cooling systems have been carefully designed for the high density provided by a full rack.

Appro can supply complete OSCAR, Rocks or Scyld clusters, or custom cluster solutions, pre-configured, including OS, compilers and clustering software, so that customers can receive a ready-to-go cluster, with their choice of storage, interconnects and networks.

Conclusion

This paper is only intended to show what is possible with an HPC Cluster, Linux and open source clustering applications. The actual applications are limited only by the imagination of the developers. With the rapid rise of Linux-based clusters, we believe it is only a matter of time until a Linux cluster is the most powerful system around due to its flexibility, manageability and low cost per FLOP. Customers using Appro HyperBlade can immediately take advantage of the flexibility provided by the Linux Operating System. Linux offers power, functionality, security and reliability for business and infrastructure applications, and is fervently supported on Appro HyperBlade.

About APPRO

High-performance Enterprise Computing - Servers, Storages, & Workstations

APPRO is a leading developer of high-performance, density-managed servers, storage subsystems, and high-end workstations for the high-performance computing and high-performance Internet computing markets. APPRO has remained as the first mover of high-performance, density-managed enterprise computing platforms to utilize the latest microprocessor technologies (both AMD and Intel) to bring cost-effective, high-performance computing systems to its target markets.

• Over 13 years of product manufacturing and engineering excellence

• Strong commitment and expertise in Linux

• Global presence - Headquarters in Milpitas, CA

Competitive Advantages:

- **Product Innovation:** Superior engineering capabilities and clever implementation of commodity parts.
- **Strategic Business and Technology Partnerships:** Raytheon, Computer Science Corporation, AMD, Intel, Infiniband Consortium and Uniwide Technologies
- **Time to Market & Customer**: Short product development cycle time and delivery through in-house engineering capabilities and technology partnerships
- **Price/Performance Value Leader:** Higher performance and quality at the lowest cost of ownership

Headquarters

Appro International
446 South Abbott Avenue
Milpitas, CA 95035
Tel: (408)941-8100
Fax: (408)941-8111
http://www.appro.com

Web Resource List

This is a list of web resources that you may find useful.

AMD Developer Center at http://www.deveopwithamd.com

The Beowulf Underground at http://beowulf-underground.org

ChaMPIon/Pro (MPI 2.1) by MSTI www.mpi-softtech.com/

Cluster Systems Management by IBM http://www-1.ibm.com/servers/eserver/clusters/software/

EnFuzion 8.0 by Axceloen www.axceleon.com/eval.html

Grid Engine http://Gridengine.sunsource.net/project/gridengine/download

The Linux HOWTO Index at http://sunsite.unc.edu/mdw/HOWTO

Linux security at http://www.luv.asn.au/overheads/security/index.html

Linux Software for Scientists at http://www.llp.fu-berlin.de/baum/linuxlist-a.html

LSF 5.1 by Platform Computing www.platform.com/products/LSFfamily/

MPICH http://www-unix.mcs.anl.gov/mpi/mpich

MyriNet Software www.myri.com/scs/

QsNET MPI Libraries
http://doc.quadrics.com/quadrics/QuadricsHome.nsf/DisplayPages/Homepage

Resource Management System by Quadrics
http://doc.quadrics.com/quadrics/QuadricsHome.nsf/DisplayPages/Homepage

Rocks Cluster Distribution www.rocksclusters.org/Rocks/

The Rocks Cluster Register www.rocksclusters.org/rocks-register

Scali MPI Connect www.scali.com/index.php

SCI Sockets by Dolphin Interconnect Solutions
www.dolphinics.com/products/software/sci_sockets.html

Scientific Applications on Linux at http://SAL.KachinaTech.COM

Scyld Beowulf Cluster Operating System www.scyld.com

Torque http://www-user.tu-chemnitz.de/~kapet/torque

Top500 site http://www.top500.org